ZeuScansion: A tool for scansion of English poetry

Manex Agirrezabal[1], *Aitzol Astigarraga*[1], *Bertol Arrieta*[1], *and Mans Hulden*[2]
[1] University of the Basque Country (UPV/EHU), Department of Computer Science, 20018 Donostia, Spain
[2] University of Colorado Boulder, Department of Linguistics, Boulder, Colorado (USA)

ABSTRACT

We present a finite-state technology (FST) based system capable of performing metrical scansion of verse written in English. Scansion is the traditional task of analyzing the lines of a poem, marking the stressed and non-stressed elements and dividing the line into metrical feet. The system's workflow is composed of several subtasks designed around finite-state machines that analyze verse by performing tokenization, part-of-speech tagging, stress placement, and stress-pattern prediction for unknown words. The scanner also classifies poems according to the predominant type of metrical foot found. We present a brief evaluation of the system using a gold standard corpus of human-scanned verse, on which a per-syllable accuracy of 86.78% is achieved. The program uses open-source components and is released under the GNU GPL license.[1]

Keywords:
scansion, English, poetry, out-of-vocabulary words

1 INTRODUCTION

Scansion is a well-established form of poetry analysis which involves marking the prosodic meter of lines of verse and possibly also dividing the lines into feet. The specific technique and scansion notation may

[1] ZeuScansion code:
`https://github.com/manexagirrezabal/zeuscansion`
Stress guesser code: `https://github.com/manexagirrezabal/athenarhythm`

http://rbc.ipipan.waw.pl

differ from language to language because of phonological and prosodic differences, and also because of different traditions regarding meter and form. Scansion is traditionally done manually by students and scholars of poetry. In the following, we present ZeuScansion, an FST-based software tool for performing this task for English poetry, and provide a brief evaluation of its performance on a gold standard corpus of poetry in various meters.

1.1 *Scansion*

Conventionally, scanning a line of poetry should yield a representation which marks every syllable with its level of stress and divides groups of syllables into units of feet. Typically two or more levels of stress are used.

Consider, for example, the following line from John Keats' poem *To autumn* (Robertson 2007, p. 137).

To swell the gourd, and plump the hazel shells

Here, a natural analysis is as follows:

```
 -   '     -   '    -     '      -  '  -    '
To swell |the gourd |and plump |the haz|el shells
```

We use the symbol ' to denote stressed (ictic) syllables, and - to denote unstressed (non-ictic) ones. That is, we have analyzed the line in question as following the stress pattern

DE-DUM DE-DUM DE-DUM DE-DUM DE-DUM

and also as consisting of five feet of two syllables each with an unstressed–stressed pattern. Indeed, this is the most common meter in English poetry: *iambic pentameter*.

The above example is rather clear-cut. How a particular line of verse *should* be scanned, however, is often a matter of contention. Consider a line from the poem *Le Monocle de Mon Oncle* by Wallace Stevens (1923):

I wish that I might be a thinking stone

Here, matters are much more murky. This line can, for example, be analyzed as five iambic feet,[2] or as one iamb, followed by a pyrrhic

[2] Iambic foot: An unstressed syllable followed by a stressed syllable [-'].

http://rbc.ipipan.waw.pl

foot,[3] followed by two stressed syllables, followed by two more iambs. The following represents several analyses of the line in question.

```
Examp.: I wish that I might be a thinking stone
1st:    -  '    -   '   -   '  -   '  -     '
2nd:    -  '    -   -   '   '  -   '  -     '
3rd:    -  '    -   '   '   '  -   '  -     '
4th:    -  '    -   -   -   '  -   '  -     '
```

The first variant is the meter most likely intended by the author. The second line represents the mentioned alternative scansion. The third and fourth lines show the output of the software tools Scandroid (Hartman 2005) and ZeuScansion, respectively.

Sometimes a line's analysis can be different from the expected one. In fact, well-known poems usually include some metrical variation; this is a stylistic device to break monotony and provide elements of surprise and variation to the reader. In the poem *The More Loving One* by W. H. Auden (Auden 1960), the poet varies the meter several times. An interesting case in point is the stanza

Admirer as I think I am
of stars that do not give a damn,
I cannot, now I see them, say
I missed one terribly all day

where the natural flow of the last line is scanned as two iambs and a double iamb.[4] While the poem itself is written in iambic tetrameters, this last line illustrates the author assigning extra emphasis on the final part: *all day*.

In short, evaluating the output of automatic scansion is somewhat complicated by the possibility of various good interpretations. As we shall see below, when evaluating the scansion task, we use a gold standard that addresses this and accepts several possible outputs as valid.

1.2 *The challenges of scansion*

Scansion is, then, the analysis of rhythmic structure in verse. But what makes it difficult? In the following, we discuss some of the immediate obstacles that have to be overcome to provide accurate annotations of rhythm and stress.

[3] Pyrrhic foot: Two unstressed syllables [- -].

[4] Double iamb: two unstressed syllables and two stressed syllables [- - ' '].

http://rbc.ipipan.waw.pl

1.2.1 Lexical stress patterns do not always apply

The primary piece of information necessary for performing metrical scansion is the *lexical stress* of words. While other elements are also important, the inherent lexical stress of a word is indispensable for the task. Consider the first line of Thomas Hardy's *The voice* (Monroe 1917, p. 131):

Woman much missed, how you call to me, call to me

If we were to simply perform scansion by marking the primary, secondary, and unstressed syllables along the line as provided for the individual words in a dictionary,[5] the result would be

```
' -   '    `       '    '  '    -  '   '    -  '
Woman much missed, how you call to me, call to me
```

This poem is in fact composed of four quatrains, where each line is written in dactylic tetrameter throughout,[6] which leads to the following analysis for this line.

```
' -   -    '       -    -  '    -  -   '    -  -
Woman much missed, how you call to me, call to me
```

As is obvious, we have to know the *prosodic stress* of the line in order to calculate the meter of the poem; simply knowing the lexical stress of each of the words will not suffice. The lexical stress is the relative emphasis inherent to certain syllables in a word, independently of the word's context. The prosodic stress shows the prominence of each of the syllables within a sentence. We address this problem by using a simplified version of some heuristics proposed by Groves (1998). Groves' rules provide a principled method to exclude some lexically stressed syllables from carrying prosodic stress.

1.2.2 Dividing the stress pattern into feet

The prosodic stress location is important, but knowledge of it is still not sufficient to obtain the intended overall meter of a poem. In order to analyze the meter, each line needs to be divided into plausible feet.

[5] We use the symbol ' to denote primary stress, the symbol ` to denote secondary stress, and - for unstressed syllables.

[6] Dactyl: a stressed syllable followed by two unstressed syllables ['--].

http://rbc.ipipan.waw.pl

A foot represents a grouping of usually one to three syllables. Returning to the above example by Thomas Hardy, we need to somehow be able to determine that the poem's lines are composed of four dactyls, and thus, that its meter is dactylic tetrameter.

In order to produce a good division of lines into feet, we employ a scoring system that takes into account not only the number of matches of the foot in the stress structure of the poem, but also the length of the feet proposed.

1.2.3 Dealing with out-of-vocabulary words

Automatic scansion is made considerably more difficult by the presence of out-of-vocabulary words. Although the lexical stress of words is not sufficient for scanning a line of poetry, it is nevertheless necessary. For some words, however, it is not available in standard dictionaries. Let us suppose that we are scanning the following line from Henry Wadsworth Longfellow's poem *The song of Hiawatha* (Longfellow 1855, p. 39):

By the shores of Gitche gumee

Here, most dictionaries would lack entries for either *Gitche* or *gumee*. For such cases, we need an informed method or algorithm for assigning lexical stress to out-of-vocabulary words. The use of rare, made-up, or unknown words is, of course, common in poetry. They appear as a result of atypical spellings, are derived through complex morphological processes, or are just nonce words coined for the occasion (cf. John Lennon's *The faulty Bagnose* or *Jabberwocky* by Lewis Carroll, 1916). Usually, the character names in poems also do not appear in dictionaries, and so their scansion cannot be inferred from such knowledge sources. This problem is exacerbated in older poetry (e.g., *Beowulf*). Failure to correctly indicate primary stress in such unknown words results in a lower accuracy of automatic scansion systems.

In order to reduce the occurrence of this type of error, we use an FST-based system that finds words spelled similarly to the target unknown word, with the assumption that their lexical stress will also be similar. More sophisticated algorithms for this purpose have been developed in Agirrezabal *et al.* (2014); such external resources can easily be embedded in ZeuScansion because of its modular design.

http://rbc.ipipan.waw.pl

2 THE OUTPUT OF ZEUSCANSION

As many different established systems of scansion exist that often vary in minor details, we have chosen a rather conservative approach, which also lends itself to a fairly mechanical, linguistic rule-based implementation. The system distinguishes three levels of stress, marks each line with a stress pattern, and attempts to analyze the predominant meter used in a poem. The following illustrates the analysis produced by our tool of a stanza from Lewis Carroll's poem *Jabberwocky* (Carroll 1916, p. 181):

```
1 He took his vorpal sword in hand:
2 Long time the manxome foe he sought-
3 So rested he by the Tumtum tree,
4 And stood awhile in thought.

1 - ' - `- ' - '
2 ' ' - `' ' - '
3 ' `- - - - `- '
4 - ' -' - '
```

In addition to this, the system also analyzes the different types of feet that make up the whole poem (discussed in more detail below). ZeuScansion supports most of the common types of foot found in English poetry, including *iambs*, *trochees*, *dactyls*, and *anapests*. Table 1 shows a complete listing of the feet supported by the tool.

Table 1: Metrical feet used in English poetry supported by ZeuScansion

	Stress pattern	Name
Disyllabic feet	- -	pyrrhus
	- '	iamb
	' -	trochee
	' '	spondee
Trisyllabic feet	- - -	tribrach
	' - -	dactyl
	- ' -	amphibrach
	- - '	anapest
	- ' '	bacchius
	' ' -	antibacchius
	' - '	cretic
	' ' '	molossus

http://rbc.ipipan.waw.pl

Once we have identified the feet used in the whole poem, we can infer the poem's meter. This includes common meters such as:

- Iambic pentameter: Lines composed of 5 iambs, used by Shakespeare in his *Sonnets* (Shakespeare 2011).
- Dactylic hexameter: Lines composed of 6 dactyls, used by Homer in the *Iliad* (Murray 1925).
- Iambic tetrameter: Lines composed of 4 iambs, used by Robert Frost in *Stopping by Woods on a Snowy Evening* (Frost 1979).

For example, if we provide Shakespeare's *Sonnets* (the whole work) as input, ZeuScansion's global analysis concludes it to be written in *iambic pentameter* (line-by-line output omitted here):

```
Syllable stress _'_'_'_'_'
Meter: Iambic pentameter
```

For Longfellow's *The song of Hiawatha*, the result of the global analysis is:

```
Syllable stress '_'_'_'_
Meter: Trochaic tetrameter
```

3 RELATED WORK

Scansion of English verse has attracted attention from numerous scholars for years. There are several books that provide general introductions to prosody in English poetry, for example, Corn (1997) or Steele (1999).

In Gerber (2013), the author compares two existing approaches to scansion: traditional stress metrics and generative metrics. In developing ZeuScansion, we have followed the traditional approach.

A number of projects also attempt to automate the scansion of English verse. Below, we give an overview of some of the current ones.

Logan (1988) documents a set of programs to analyze sound and meter in poetry. This work falls in a general genre of techniques that attempt to analyze the phonological structure of poems following the generative phonological theory outlined by Chomsky and Halle (1968) and described by Brogan (1981).

Scandroid is a program that scans English verse written in either iambic or anapestic meter, designed by Charles O. Hartman (1996;

http://rbc.ipipan.waw.pl

2005). The source code is publicly available.[7] The program can analyze poems and check if the predominant stress pattern is iambic or anapestic. However, if the input poem's meter is not one of those two, the system forces each line into one of them.

AnalysePoems is another tool for automatic scansion and identification of metrical patterns written by Marc Plamondon (2006). In contrast to Scandroid, AnalysePoems only identifies patterns; it does not impose them. The program also checks the rhyme scheme found in the input poem. It is reportedly developed in *Visual Basic* and the .NET framework; however, neither the program nor the code appear to be available.

Calliope is a similar tool, built on top of Scandroid by Garrett McAleese (2007). It is an attempt to take advantage of linguistic theories of stress assignment in scansion. The program does not seem to be freely available.

Of the current efforts, Greene *et al.* (2010) appears to be the only one that uses statistical methods in the analysis of poetry. For the learning process, *The Sonnets* by Shakespeare was used, as well as a number of other works freely available online.[8] Weighted finite-state transducers were used for stress assignment. As with the other documented projects, we have not obtained an implementation to review.

4 CORPORA

Several different corpora were used for the development of ZeuScansion. These include the pronunciation dictionaries NETtalk (Sejnowski and Rosenberg 1987) and CMU (Weide 1998), which both list pronunciations of words, the number of syllables they contain, as well as indications of primary and secondary stress location. Each employs a slightly different notation, but they are in general quite similar in content as they both mark three levels of stress and show pronunciations:

```
NETTALK format:
@bdIkeS|n     `_'_     S4     abdication     0     (N)

CMU format:
INSPIRATION    IH2 N S P ER0 EY1 SH AH0 N
```

[7] http://oak.conncoll.edu/cohar/Programs.htm

[8] http://www.sonnets.org

http://rbc.ipipan.waw.pl

We also use a human-annotated poetry corpus obtained from an interactive learning environment program for training people to scan traditionally metered English poetry called For Better For Verse (Tucker 2011).[9] The poems on the site are marked up with TEI P5 coding, a convenient format for poetry markup.[10] The collection of poems is rather homogeneous, the predominant meter of the poems being iambic (92.7% of the lines). The remaining 7.3% lines use trochaic (3.65%), anapestic (2.09%) or dactylic (1.56%) meters. We employ this corpus in order to evaluate the performance of ZeuScansion.

In addition to this source, we downloaded several poems from Project Gutenberg (Hart 1971) for evaluation and testing purposes.[11]

Finally, we used the Wall Street Journal section of the Penn Treebank (Marcus *et al.* 1993) to train a part-of-speech-tagger, the role of which is described below.

5 METHOD

Our tool is constructed around a number of guidelines for scansion developed by Peter L. Groves (1998). It consists of three main components:

(a) an implementation of Groves' rules of scansion – mainly a collection of POS-based stress-assignment rules,

(b) a pronunciation lexicon together with an out-of-vocabulary word-stress guesser, and

(c) a 'plausible foot division' system.

5.1 *Groves' rules*

Groves' rules try to assign stress levels in a way that turns this task, as far as possible, into an objective process driven by lexicon and syntax, independent of more elusive concepts of the poem such as meaning and intent. The rules assign stress as follows:

1. Primarily stressed syllables of content words (nouns, verbs, adjectives, and adverbs) receive **primary stress**.

[9] `http://prosody.lib.virginia.edu`

[10] `http://www.tei-c.org/release/doc/tei-p5-doc/en/html/VE.html`

[11] `http://www.gutenberg.org`

http://rbc.ipipan.waw.pl

2. Secondarily stressed syllables in polysyllabic content words, primarily stressed syllables in polysyllabic function words (auxiliaries, conjunctions, pronouns, and prepositions) and secondarily stressed syllables in compound words get **secondary stress**.
3. Unstressed syllables of polysyllabic words and monosyllabic function words are **unstressed**.

In Section 6 we present a more elaborate example to illustrate how Groves' rules are implemented.

5.2 *Pronunciation lexicon and out-of-vocabulary word-stress guesser*

To calculate the basic stress pattern of words necessary for Groves' rules, we mainly use the dictionaries mentioned earlier: the CMU pronunciation dictionary and NETtalk. The system first attempts to locate the stress pattern in the smaller NETtalk dictionary (20,000 words) and then falls back to using CMU (125,000 words) if the word is missing in NETtalk. The merged lexicon, where NETtalk pronunciations are given priority, contains about 133,000 words.

In the event that a word cannot be found in either the NETtalk lexicon or the CMU dictionary, we try to guess the stress pattern of the word using an FST-based system, which relies on the hypothesis that similarly spelled words have the same stress pattern.

5.3 *Foot division system*

The final subtask – no less important than the previous ones – is to divide a line's stress pattern into feet, for which we use a scoring system. The scoring system takes two features into account: the number of matches that each possible foot has in the line and the number of syllables that that foot has. More details are given below.

6 ZEUSCANSION: TECHNICAL DETAILS

The structure of the system is divided into the subtasks shown in Figure 1. We begin with preprocessing and tokenization, followed by part-of-speech tagging. Then, we find the lexical stress pattern for each word, guessing the stress patterns for any words not found in the dictionary. After these preliminaries, we apply Groves' scansion rules to determine the prosodic stress and perform some cleanup of the result.

http://rbc.ipipan.waw.pl

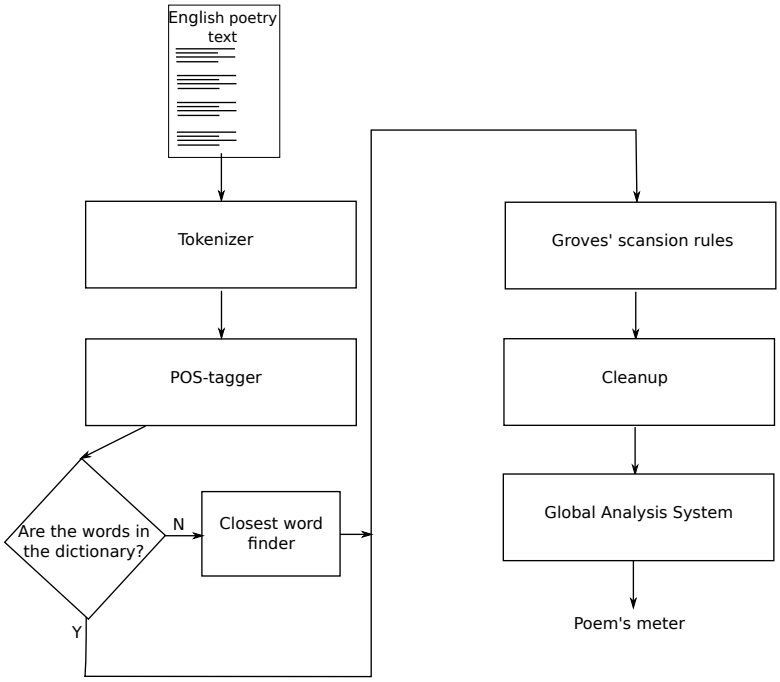


Figure 1: Structure of ZeuScansion

Finally, we calculate the average line stress pattern, which we later try to divide into feet.

The toolchain is implemented as a chain of finite-state transducers, each of them written using the *foma* toolkit (Hulden 2009),[12] save for the part-of-speech tagger, which is a Hidden Markov Model (HMM) implementation (Halácsy *et al.* 2007). We use *Perl* as a glue language to communicate between the components.

6.1 *Preparation of the input data*

After tokenization,[13] we obtain the part-of-speech (POS) tags of the words of the poem. For the POS tagger, we trained Hunpos[14] (Halácsy *et al.* 2007) on the Wall Street Journal section of the Penn Treebank (Marcus *et al.* 1993). While other, more general, corpora might be more suitable for this task, we only need to distinguish between func-

[12] `https://foma.googlecode.com`

[13] Code available at `https://code.google.com/p/foma/wiki/FAQ`.

[14] `https://hunpos.googlecode.com`

http://rbc.ipipan.waw.pl

tion and non-function words, and thus performance differences would most likely be slight between tagger implementations.

Once the first process is completed, the system starts applying Groves' rules. This process is also encoded as finite-state transducers. To apply the rules, however, we must know the stress pattern of each word. Here, as mentioned above, we resort to a heuristic for assigning lexical stress to out-of-vocabulary words.

The strategy we use to analyze such words is to find a close neighboring word in the dictionary, relying on an intuition that words that differ very little in spelling from the sought-after word are also likely to be pronounced in a similar way, or, at the very least, exhibit the same stress pattern.

6.2 *Finding the closest word*

In order to find what we call the *closest word* in the dictionary, we construct a cascade of finite-state transducers from the existing dictionaries in such a way that, given an input word, it will output the most similar word, according to spelling, using a metric of word distances that we have derived for the purpose. These transducers will perform small specific changes (substitution, insertion, and deletion) on the input word, such as:

- change one vowel,
- change one consonant,
- change two vowels,
- change one vowel and one consonant,
- change two consonants.

Before performing any of these changes, we divide the unknown word into two parts, where the second part represents roughly the last syllable. Then, we perform the aforementioned changes in each part of the word. If, when performing any one of those changes, we find an existing word, the system will return that word and not proceed with the other changes. For example, in the following line from Shakespeare's *Romeo and Juliet* (Shakespeare 1806, p. 77)

And ***usest*** *none in that true use indeed*

we find the word *usest* (the archaic second person singular, simple present form of the verb *use*), which does not appear in our lexicon.

http://rbc.ipipan.waw.pl

Our closest-word finder begins with the word splitter, which would return `u|sest`. Then, it maps this word to all possible words produced by changing just one vowel in the first part of the word, one vowel in the second part, or changing one consonant. In this example case, after performing some of these changes, the system would determine the closest match according to the scheme above to be *wisest*, and assume that its lexical stress matches that of *usest*. This is achieved by changing one vowel and inserting a consonant at the beginning of the word.

These transducers need to be correctly ordered, as an earlier transducer in the cascade will have priority over later ones. In our cascade, the dictionaries are also included as the very first mapping. If the word is not found in the dictionary, subsequent transducers perform the various mappings, filtering their outputs in such a way as to be constrained against possible words in the dictionary. The actual order in the cascade was determined based on the precision achieved in cross-validation against the NETtalk dictionary. To illustrate this, consider a pair of transducers, one performing just one vowel change and the other changing only one consonant. If the first transducer can guess the correct word stress in, say, 90% of the cases and the other one in 10% of the cases, we order the vowel transducer first in the cascade, and the consonant transducer second. In the case that a close word is not found making the possible mentioned changes, the finder will return the symbol `?` as a result.

6.3 *Implementation of Groves' rules*

Once we have obtained the lexical stress for each word, we employ a finite-state transducer that encodes each step in Groves' rules in replacement rules (Beesley and Karttunen 2003).

Groves' rules dictate that the primarily stressed syllable in content words will maintain primary stress. In polysyllabic function words, the syllable carrying primary lexical stress will be assigned secondary stress. Secondary stresses in polysyllablic content words will maintain secondary stress. All other syllables will be unstressed.

The input for these transducers is a string with the structure `word+POS`. The output is the stress pattern of the word after apply-

http://rbc.ipipan.waw.pl

ing Groves' rules, written as `word+stress+POS`. Let's consider a line from Longfellow's poem *The song of Hiawatha*:

*changed them thus **because** they mocked you*

For an analysis of the word *because*, the input for the transducer that encodes Groves' rules would be `because+IN`. The lexical resources transducer would locate the word in the dictionary and determine that the second syllable carries primary stress while the first syllable is unstressed. After applying the prosodic stress rules, the system would return that the second syllable should receive secondary stress (instead of the original primary) as the input word is a polysyllabic function word. Hence, the output of the transducer would in this case be `because+-`+IN`.

The last step is to remove all the material not strictly required for working with stress patterns. For the cleanup process, we use a transducer that removes everything before the first + character and everything after the second + character. It then removes all the + characters, so that the only result we get is the bare stress structure of the input word:

`because+-`+IN` → `-``

6.4 *Global analysis*

After the stress rules have been applied and we know the stress levels of each syllable of each line, we move to the meter inference process. To this end, we calculate the entire poem's average stress structure. This is encoded by a vector of syllable positions. Each line is examined and for each syllable and its position we add numerical values depending on the syllable's stress. The pseudocode of the average stress calculator is as follows:

```
vector[1..nsylls]=0
foreach line (1..nlines) {
   foreach syllable (1..nsylls) {
      if stress(syllable) == '
          vector[syllable] = vector[syllable] + 2
      if stress(syllable) == `
          vector[syllable] = vector[syllable] + 1
   }
}
```


http://rbc.ipipan.waw.pl

We illustrate the process with the following excerpt from *The song of Hiawatha* as the input (Longfellow 1855, p. 146):

Barred with streaks of red and yellow$_1$
Streaks of blue and bright vermilion$_2$
Shone the face of Pau-Puk-Keewis$_3$
From his forehead fell his tresses$_4$
Smooth and parted like a woman's$_5$
Shining bright with oil and plaited$_6$
Hung with braids of scented grasses$_7$
As among the guests assembled$_8$
To the sound of flutes and singing$_9$
To the sound of drums and voices$_{10}$
Rose the handsome Pau-Puk-Keewis$_{11}$
And began his mystic dances$_{12}$

According to Groves' rules, the stress values for each line are:

```
'-`-'-'`1
`-'-'-'-2
  '-'-?3
--'`'-`-4
'-`---`-5
`-'-'-`-6
'-`-`-`-7
'-`-`-`-8
--'-`-`-9
--'-`-`-10
  '-'-?11
--`-'-`-12
```

Our algorithm would then calculate the following:

Syllable	1	2	3	4	5	6	7	8
Σ	14	0	19	1	14	0	12	1
Normalized	0.74	0	1	0.05	0.74	0	0.63	0.05
Stress	'	-	'	-	'	-	'	-

These numbers represent each syllable's average stress over the entire poem. In Figures 2 and 3 we show a graphical representation

http://rbc.ipipan.waw.pl

Figure 2:
Average stress level per syllable position in Shakespeare's *Sonnets*

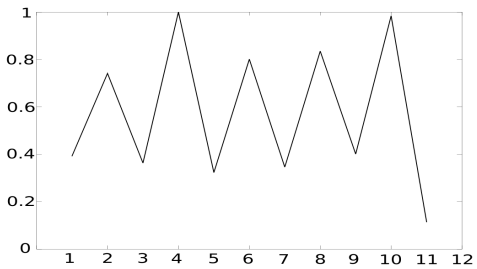
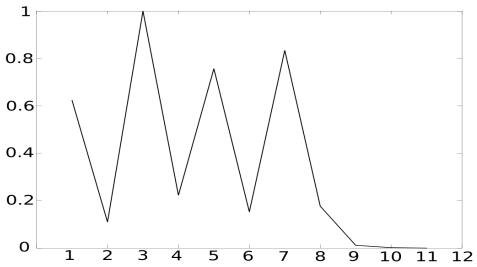


Figure 3:
Average stress level per syllable position in Longfellow's *The song of Hiawatha*



of these numbers based on an analysis of Shakespeare's *Sonnets* and Longfellow's *The song of Hiawatha*. We use 0.5 as a cutoff value: if the normalized average stress for a syllable is greater than this, it is assigned the label *stressed* and otherwise *unstressed*. We assume that all the lines contain the same number of syllables. This naturally leads to difficulties with works with differing syllable counts per line (such as *Phantasmagoria* and other poems by Lewis Carroll, 1869). We set aside the interesting problems surrounding proper normalization and treatment of mixed-line poems for future work.

After the above steps, we attempt to divide the average stress pattern into feet with the goal of producing a global analysis of the poem. In our previous example, it is obvious that the optimal meter to assign is trochaic tetrameter, a sequence of four trochees, but in other cases foot-division can be ambiguous. Consider, for instance, the meter:

```
-'--'--'--'-
```

which could be analyzed as consisting mainly of (1) amphibrachs [- ' -], (2) trochees [' -] and (3) iambs [- ']. All three patterns appear four times in the line. For such cases, we have elaborated a scoring system for selecting the appropriate pattern: we give a weight of 1.0

http://rbc.ipipan.waw.pl

for hypothetical disyllabic patterns, and a weight of 1.5 for trisyllabic ones. In this example, this would yield the judgement that the structure is amphibrachic tetrameter (1.5 × 4 matches = 6). This example is illustrated in Table 2.

Foot	Pattern	N¬∫ matches	Score
Amphibrach	-'-	4	6
Iamb	-'	4	4
Trochee	'-	4	4
Anapest	'--	3	4.5
Dactyl	'--	3	4.5
Pyrrhus	- -	3	3

Table 2:
Hypothetical feet for the meter -'--'--'--'-

We also attempted to develop an alternative foot-division strategy by taking into account how many syllables were omitted in the analysis. For example, in the previous Longfellow example at line 12, the system would note two unused syllables. The intuition was that a collection of feet that left less unaccounted syllables should be the preferred meter. After evaluating this procedure, however, the results were consistently lower than with the first-mentioned scoring system, which we then chose to use.

7 FURTHER EXPLORATIONS

In the preceding section, we have presented ZeuScansion, an implemented system for scansion, available online.[15] However, we also explored possible improvements for its out-of-vocabulary word-stress guesser. To this end, we developed two alternative approaches based on linguistic generalizations and machine learning techniques. In this section, we will outline how these two systems assign stress to out-of-vocabulary words. While the system described earlier also assigned secondary stresses to words, the alternatives only produce a prediction of the placement of primary stress. However, once the primary stress is assigned, predicting the location of secondary stresses is quite straightforward.

These systems receive a word as input and return the location of primary stress. For example, with the word *introduction* as input, these

[15] https://github.com/manexagirrezabal/zeuscansion

http://rbc.ipipan.waw.pl

guessers should return `--'-`, given that the primary stress is located in the third syllable (*duc*).

Our ultimate goal is to include the best one out of all these approaches in the final ZeuScansion implementation. The source code for these stress assignment tools is made available under the GNU GPL license.[16]

7.1 *Linguistic approach*

For the linguistic approach we have programmed a linguistic toolchain that performs grapheme-to-phoneme conversion (G2P), syllabification and stress assignment.

We first convert the orthographic representation of words to sequences of phonemes, using a G2P system presented in Novak *et al.* (2012).[17] Following this, we syllabify the words using a finite-state syllabification algorithm (Hulden 2006). Our main concern for stress assignment is the weight of the syllables, which might be light or heavy, captured as follows:

- Heavy syllable: The syllable has a coda or ends in a tense vowel.
- Light syllable: Any syllable not classified as heavy.

After this processing, we apply several stress assignment rules that rely on various linguistic generalizations regarding the English vocabulary. The main active rule is the so-called *Latin stress rule* (Halle and Vergnaud 1987), which, despite the name, also applies to many English polysyllabic words. This rule codifies the generalization that heavy syllables tend to attract stress. Below is a description of this, divided into four subrules:

- If the penultimate syllable is light, the antepenultimate syllable is stressed.
- If the penultimate syllable is heavy, it is stressed.
- In the case of disyllabic words, the first syllable is stressed.
- Monosyllabic words are stressed.

Despite the descriptive power of the generalization, examples exist of words where it fails, such as *an*|*té*|*nna*, *a*|*la*|*bá*|*ma* or *po*|*líce*.

[16] `https://github.com/manexagirrezabal/athenarhythm`

[17] `https://github.com/AdolfVonKleist/Phonetisaurus/`

http://rbc.ipipan.waw.pl

7.2 *Machine learning approach*

In our machine learning approach we have trained a Support Vector Machine (SVM) (Chang and Lin 2011; Fan *et al.* 2008) using the NETTalk stress-annotated dictionary. We treat the stress assignment task as a multi-class classification problem. The class to be assigned is the stress pattern that each word follows, taking into account only the main stress. We extracted 25 different stress patterns from our dictionary, where each stress pattern is a sequence of symbols for stressed and unstressed syllables (' and -).

We used two different sets of features for the purpose of training the SVMs. In the first set, FS1, we used character bigrams as features, including word boundaries as a special character. In the second feature set, FS2, we used character trigram frequencies, also known as *Wickelfeatures* (Rumelhart and McClelland 1985). For example, given the word *reference*, with FS1 we would train the SVM with the information that the bigrams {#r}, {ef}, {fe}, {er}, {en}, {nc}, {ce}, {e#} appeared once, the bigram {re} twice and all other possible bigrams zero times. These, together with the length of the word, are the training features for the first set. In the second feature set, we include the frequencies of trigrams, in this case {#re}, {ref}, {efe}, {fer}, {ere}, {ren}, {enc}, {nce}, {ce#}. For the example word *reference*, the correct class would be ' -, indicating that the first syllable carries primary stress.

Naturally, these features need to be encoded as numbers; a simple mapping function performs this mapping. After this, we produced a corpus of 19,528 instances, one instance per word in the dictionary. In the case of FS1, each word was represented using 899 feature–value pairs, while in FS2, 5,495 feature–value pairs were required.

The feature set that yielded the highest performance using cross-validation over the training set was the set consisting of character bigrams and their frequencies (FS1). We trained different support vector machines with varying parameters. The best performing one was a Support Vector Classifier using an RBF/Gaussian kernel, whose parameters C and γ (soft-margin penalty and the Gaussian dispersion) were tuned by a grid-search.

http://rbc.ipipan.waw.pl

EVALUATION

As the gold standard material for evaluation, we used the corpus of scanned poetry For Better For Verse, made available by the University of Virginia, from which we extracted the reference analyses. Sometimes several analyses are given as correct. The results of the evaluation are given in Table 3. 86.78% of syllables are scanned correctly in the best configuration of ZeuScansion. This is slightly below the performance of Scandroid per syllable. As our test corpus is mainly iambic, Scandroid of course has an advantage in that it is fixed to only handle iambic or anapestic feet.

Table 3: ZeuScansion evaluation results against the For Better For Verse corpus

	Scanned lines	Correctly scanned	Accuracy
ZeuScansion	759	199	26.21%
Scandroid	759	326	42.95%

	Scanned sylls.	Correctly scanned	Accuracy
ZeuScansion	7076	5999	86.78%
Scandroid	7076	6353	89.78%

We evaluate our system by checking the error rate obtained by using Levenshtein distance comparing ZeuScansion's output for each line of the analyzed poem against the gold standard scansion. We do this in order not to penalize missing or superfluous syllables, which are sometimes present, with more than 1 count. For example, this line from Longfellow's poem *The song of Hiawatha*,

sent the wildgoose wawa northward

written in trochaic tetrameter, should be scanned as

```
'-'-'-'-
```

while our tool marks the line in question as

```
'-?'-'-
```

after conversion to using only two levels of stress from the original three-level marking. For the conversion, we consider primarily and secondarily stressed syllables stressed, and unstressed syllables unstressed. With the Levenshtein metric we evaluate the distance between the analysis proposed by our tool, ZeuScansion, and the gold

http://rbc.ipipan.waw.pl

Poem	Correctly classified
The song of Hiawatha	32.03%[18]
Shakespeare's Sonnets	70.13%

Table 4:
Evaluation of the global analysis system (only ZeuScansion)

standard. Obviously, any proposed analysis identical to the gold standard will be assigned a distance of zero. The value that we obtain from using this distance metric can be interpreted as a minimum number of errors in the analysis. In the example, ZeuScansion fails to assign the correct stress pattern to *wildgoose*, because the word does not appear in dictionaries and no similarly spelled word can be found. The minimum Levenshtein distance between the analysis and the reference is two, since changing the third `?` to a `'` and adding a `-` to the analysis would produce the stress pattern given for this line in the gold standard.

We also evaluated the global analysis system using two different works of poetry. The first one is Longfellow's *The song of Hiawatha* and the second one Shakespeare's *Sonnets*. We analyzed the meter of each sonnet in Shakespeare's writing (154 sonnets); in the case of Longfellow's poem we analyzed each stanza (637 stanzas) separately. Shakespeare's sonnets are written in iambic pentameter and *The song of Hiawatha* in trochaic tetrameter. Table 4 reports the accuracy on this task.

8.1 *Out-of-vocabulary word-stress guesser*

Since the out-of-vocabulary word-stress guesser impacts on the overall quality of the system, we have evaluated that component separately. ZeuScansion only uses the similarity approach for the out-of-vocabulary word-stress guessing process. However, we intend to include the linguistic and machine learning approaches in the future as they achieve better results.

The NETtalk pronunciation dictionary was used for evaluating this phase. As some of the methods for stress assignment are data-driven and others not, we evaluated them slightly differently. Both the similarity approach and machine learning approach were evaluated using 10-fold cross-validation. The linguistic approach, however, was evaluated against the whole corpus without any splitting, as it does

[18] 44.58% were classified as amphibraic dimeter.

http://rbc.ipipan.waw.pl

Table 5:
Evaluation results for the out-of-vocabulary word-stress guesser

	Accuracy
FST-based approach	67.77%
Linguistic approach	**73.62%**
Machine Learning approach	70.98%

not rely on any training data and is essentially an expert system. The results are shown in Table 5.

The highest accuracy is achieved by the linguistic generalization; however, both the results for using SVMs and those for using hand-encoded generalizations are sufficiently close to warrant further research in the improvement of both.

9 DISCUSSION AND FUTURE WORK

In this article, we have presented a basic system for scansion of English poetry. The evaluation results are promising: a qualitative analysis of the remaining errors reveals that the system, while still containing errors vis-à-vis human expert judgements, makes very few egregious errors. We expect to develop the system further in several respects.

We intend to apply new stress-guessing algorithms in ZeuScansion that yield better results. We believe that the general results of the system will improve slightly.

We also plan to add statistical information about the global properties of poems to resolve uncertain cases in a manner consistent with the overall structure of a given poem. Such additions could resolve ambiguous lines and try to make them fit the global pattern of a poem. What we have in mind is the replacement of the part-of-speech tagging process by a deterministic FST-based tagger such as Brill's tagger (Roche and Schabes 1995). This would allow the representation of the entire tool as a single finite-state transducer composed of several subparts. Under such a model, however, we would not be able to use other word-stress guessing algorithms than the similarity approach. In the short term, we also expect to tackle improvements regarding the possibility of analyzing mixed-length lines.

We believe that the availability of a gold-standard corpus of expert scansion offers a valuable improvement in the quantitative assessment of the performance of future systems and modifications.

http://rbc.ipipan.waw.pl

As noted in Agirrezabal *et al.* (2014), there is still room for improvement in the out-of-vocabulary word-stress allocation systems. One of the main issues is the addition of information about the part of speech to the learning corpus. This is necessary because disyllabic words, which are quite frequent, tend to behave differently along the lines of noun–verb distinction.[19] We believe that with this improvement the accuracy of the linguistic and the machine learning paradigm might see significant gains in accuracy.

To conclude, as our main research project involves both poetry analysis and generation, we intend to use this implementation in the generation of poetry using morphosyntactic patterns following the philosophy of Agirrezabal *et al.* (2013).

ACKNOWLEDGMENTS

The first author's work is funded by a PhD grant from the University of the Basque Country and supported by the Association of the Friends of Bertsolaritza. We are also grateful to the *University of Delaware*, where part of this work was undertaken in the *Department of Linguistics and Cognitive Sciences*. We especially acknowledge the help of professor Jeffrey Heinz. The help of Herbert Tucker, author of the For Better For Verse project, has been fundamental to evaluate our system. We also want to extend thanks to the Scholar's Lab, an arm of the University of Virginia Library, maintainers of For Better For Verse. Joseph Gilbert and Bethany Nowviskie have been most steadily helpful there.

REFERENCES

Manex Agirrezabal, Bertol Arrieta, Aitzol Astigarraga, and Mans Hulden (2013), POS-tag based poetry generation with WordNet, *Proceedings of the 2013 European Workshop on Natural Language Generation*, pp. 162–166.

Manex Agirrezabal, Jeffrey Heinz, Mans Hulden, and Bertol Arrieta (2014), Assigning stress to out-of-vocabulary words: three approaches, *International Conference on Artificial Intelligence*, 27:105–110.

Wystan H. Auden (1960), The more loving one, `https://www.poets.org/poetsorg/poem/more-loving-one`.

[19] If the word is a noun, the stress goes on the first syllable, e.g., *récord*. If it is a verb, the second syllable is stressed, e.g. *recórd*.

http://rbc.ipipan.waw.pl

Kenneth R. Beesley and Lauri Karttunen (2003), Finite-state morphology: Xerox tools and techniques, *CSLI*.

Terry V. F. Brogan (1981), *English versification, 1570-1980: a reference guide with a global appendix*, Johns Hopkins University Press.

Lewis Carroll (1869), *Phantasmagoria and Other Poems*, London : Macmillan, https://archive.org/details/phantasmagoriaot00carrrich.

Lewis Carroll (1916), *Alice's Adventures in Wonderland and Through the Looking Glass*, George W. Jacobs & Company, Philadelphia.

Chih-Chung Chang and Chih-Jen Lin (2011), LIBSVM: A library for support vector machines, *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Noam Chomsky and Morris Halle (1968), *The sound pattern of English*, Harper & Row.

Alfred Corn (1997), *The poem's heartbeat: a manual of prosody*, Copper Canyon Press.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin (2008), LIBLINEAR: A library for large linear classification, *The Journal of Machine Learning Research*, 9:1871–1874, software available at http://www.csie.ntu.edu.tw/~cjlin/liblinear.

Robert Frost (1979), *The poetry of Robert Frost: the collected poems, complete and unabridged*, Henry Holt and Company.

Paul Fussell (1965), *Poetic meter and poetic form*, McGraw Hill.

Natalie Gerber (2013), Stress-based metrics revisited: a comparative exercise in scansion systems and their implications for iambic pentameter, *Thinking Verse*, III:131–168.

Erica Greene, Tugba Bodrumlu, and Kevin Knight (2010), Automatic analysis of rhythmic poetry with applications to generation and translation, in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pp. 524–533.

Peter L. Groves (1998), *Strange music: the metre of the English heroic line*, volume 74 of *ELS Monograph Series*, English Literary Studies, University of Victoria.

Péter Halácsy, András Kornai, and Csaba Oravecz (2007), HunPos: an open source trigram tagger, in *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (Interactive Poster and Demonstration Sessions)*, pp. 209–212.

Morris Halle and Jean-Roger Vergnaud (1987), *An essay on stress*, MIT Press.

Michael Hart (1971), Project Gutenberg, https://www.gutenberg.org/.

http://rbc.ipipan.waw.pl

Charles O. HARTMAN (1996), *Virtual muse: experiments in computer poetry*, Wesleyan University Press.

Charles O. HARTMAN (2005), The Scandroid 1.1, `http://oak.conncoll.edu/cohar/Programs.htm`.

Mans HULDEN (2006), Finite-state syllabification, in Anssi YLI-JYRÄ, Lauri KARTTUNEN, and Juhani KARHUMÄKI, editors, *Finite-State Methods and Natural Language Processing*, volume 4002 of *Lecture Notes in Computer Science*, pp. 86–96, Springer.

Mans HULDEN (2009), Foma: a finite-state compiler and library, in *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics (Demonstrations Session)*, pp. 29–32.

Harry M. LOGAN (1988), Computer analysis of sound and meter in poetry, *College Literature*, 15(1):19–24.

Henry W. LONGFELLOW (1855), *The Song of Hiawatha*, David Bogue, `https://archive.org/details/songhiawathathe00longrich`.

Mitchell P. MARCUS, Mary Ann MARCINKIEWICZ, and Beatrice SANTORINI (1993), Building a large annotated corpus of English: the Penn Treebank, *Computational Linguistics*, 19(2):313–330.

William G. M. MCALEESE (2007), Improving scansion with syntax: an investigation into the effectiveness of a syntactic analysis of poetry by computer using phonological scansion theory, Technical report, Department of Computing Faculty of Mathematics, Computing and Technology The Open University.

Harriet MONROE (1917), *The new poetry: an anthology*, The Macmillan Company, `https://archive.org/details/newpoetryananth01hendgoog`.

Augustus T. MURRAY (1925), *Homer: The Iliad*, Heinemann, `https://archive.org/details/iliadmurray01homeuoft`.

Josef NOVAK, Nobuaki MINEMATSU, and Keikichi HIROSE (2012), WFST-based grapheme-to-phoneme conversion: open source tools for alignment, model-building and decoding, in *Proceedings of the 10th Internaltional Workshop on Finite-State Methods and Natural Language Processing*, pp. 45–49.

Marc R. PLAMONDON (2006), Virtual verse analysis: analysing patterns in poetry, *Literary and Linguistic Computing*, 21(1):127–141.

Margaret ROBERTSON, editor (2007), *Poems published in 1820 by John Keats*, Project Gutenberg, `http://www.gutenberg.org/ebooks/23684`.

Emmanuel ROCHE and Yves SCHABES (1995), Deterministic part-of-speech tagging with finite-state transducers, *Computational Linguistics*, 21(2):227–253.

David E. RUMELHART and James L. MCCLELLAND (1985), On learning the past tenses of English verbs, Technical report, Institute for Cognitive Science, University of California, San Diego.

http://rbc.ipipan.waw.pl

Terrence J. SEJNOWSKI and Charles R. ROSENBERG (1987), Parallel networks that learn to pronounce English text, *Complex Systems*, 1(1):145–168.

William SHAKESPEARE (1806), *Romeo and Juliet*, John Cawthorn.

William SHAKESPEARE (1904), *The tragedy of Hamlet*, Cambridge University Press.

William SHAKESPEARE (2011), *Shakespeare's sonnets*, Project Gutenberg, `https://archive.org/details/shakespearessonn01041gut`.

Timothy STEELE (1999), *All the fun's in how you say a thing: an explanation of meter and versification*, Ohio University Press.

Wallace STEVENS (1923), *Harmonium*, Academy of American Poets.

Herbert F. TUCKER (2011), Poetic data and the news from poems: a *For Better for Verse* memoir, *Victorian Poetry*, 49(2):267–281.

Robert L. WEIDE (1998), *The CMU pronunciation dictionary, release 0.6*, Carnegie Mellon University.

This work is licensed under the Creative Commons Attribution 3.0 Unported License.
`http://creativecommons.org/licenses/by/3.0/`

http://rbc.ipipan.waw.pl